AL-TP-1991-0011

ARMSTRONG LABORATORY

AD-A235 920

NEURAL NETWORKS: A PRIMER

Vince L. Wiggins

RRC, Incorporated
3833 Texas Avenue, Suite 256
Bryan, TX 77802

Larry T. Looper
Sheree K. Engquist, Capt, USAF

HUMAN RESOURCES DIRECTORATE
MANPOWER AND PERSONNEL DIVISION
Brooks Air Force Base, TX 78235-5000

May 1991

Final Technical Paper for Period September 1989 – March 1991

Approved for public release; distribution is unlimited.

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5000

NOTICES

This technical paper is published as received and has not been edited by the technical editing staff of the Armstrong Laboratory.

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Office of Public Affairs has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This paper has been reviewed and is approved for publication.

LARRY T. LOOPER
Project Scientist

WILLIAM E. ALLEY, Technical Director
Manpower and Personnel Division

MICHAEL W. BIRDLEBOUGH, Colonel, USAF
Chief, Manpower and Personnel Division

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
May 1991

3. REPORT TYPE AND DATES COVERED
Final Paper – September 1989 – March 1991

4. TITLE AND SUBTITLE
Neural Networks: A Primer

5. FUNDING NUMBERS
C – F41689-88-D-0251
PE – 62205F
PR – 7719
TA – 20
WU – 20

6. AUTHOR(S)
Vince L. Wiggins
Larry T. Looper
Sheree K. Engquist

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
RRC, Incorporated
3833 Texas Avenue, Suite 256
Bryan, TX 77802

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)
Armstrong Laboratory
Human Resources Directorate
Manpower and Personnel Division
Brooks Air Force Base, TX 78235-5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AL-TP-1991-0011

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Neural network technology has recently demonstrated capabilities in areas important to personnel research such as statistical analysis, decision modeling, control, and forecasting. This paper is an introduction to neural network terminology, definitions, capabilities, and applications. It provides a discussion of traditionally used statistical methods and compares them to neural network architectures. The benefits of using neural networks is their ability to derive nonlinear and interacting relationship among variables of a model. This paper is designed for individuals who have little or no knowledge of this growing field.

14. SUBJECT TERMS
back propagation
classification problem-solving
neural networks
personnel system modeling

15. NUMBER OF PAGES
42

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

TABLE OF CONTENTS

LIST OF FIGURES

PREFACE

This technical paper is the first publication in an area of research, neural network applications, under the Manpower and Personnel Division's Force Management program. Development of this technology in the personnel modeling arena will greatly improve the capability to understand the interdependencies of many related variables

The authors wish to thank Ms Kathy Berry for assistance in reviewing and preparing this document. Dr Brice M. Stone and Dr Thomas R. Saving provided many interesting views on the material.

SUMMARY

This technical paper introduces the concepts of neural networks with emphasis on the Air Force personnel system. Neural networks offer a method of analyzing and simulating the personnel system with few restrictions on the form of the relationships in the system. The system can be estimated or "trained" with all of its interdependencies considered. It provides a basic foundation on which further research can be done. The paper is an introductory primer designed for individuals who have little or no knowledge of this growing field. Additional information on the specific applications of neural networks within the field of manpower and personnel can be found in the final report titled, *Neural Networks and their Application to Air Force Personnel Modeling*, published by the Manpower and Personnel Division of the Air Force Human Resources Laboratory, currently in press.

I. INTRODUCTION

This technical paper provides an explanation of neural network technology in the form of a primer. The primer gives a basic understanding of how neural networks are used, what they are capable of, and some implementation details.

The second section is a brief history to introduce the scope of the field and discuss why there has been a sudden push of interest in the field. In Section III, a working definition of neural networks, network capabilities, and some real world applications are addressed. In Section IV, traditional methods, such as logit analysis, are compared to the use of neural networks for the same application. Section V examines two of the many types of network architectures to give an understanding of the learning techniques most commonly applied in neural network research. Section VI provides example applications in the area of personnel modeling.

II. HISTORY

The ideas immanent in neural networks have been around for a long time. As early as 1890 William James, in his psychology primer, laid out many of the general concepts still used in neural network research. His treatment was purely conceptual and little was done to extend the models he outlined. In 1943, Warren McCullouch and Walter Pitts created the first formal models of neurons and neural networks, and this launched a series of more extensive explorations of neural networks. For most of the researchers, the capabilities of biological systems were primary motivating factors. Many researchers hoped to emulate the capabilities of the brain and nervous system by creating systems based on what was known about real neurons and their network structure. At that time, neural networks were even viewed as alternatives to digital computers in creating automated systems.

During 1969 Marvin Minsky and Seymour Papert published an influential book that took the bloom off this early research. In *Perceptrons*, they rigorously proved a basic limitation of the primary class of neural network models that were being studied at the time—they were linear. This meant the models could not solve problems that were not linearly separable. Moreover, because of this limitation, these models of neurons and neural networks could not be combined to produce general computing engines. In short, there were problems they simply could not solve. Despite possibilities for extending the models Minsky and Papert had analyzed, the impact of their publication was significant.

During the 1970's and early 80's less research was done in the field, and much of this research was performed in Europe and Asia. The majority of researchers were neurobiologists and mathematicians. In the early 1980's several elements converged to generate an explosion of interest in the field.

Four major factors contributed to the resurgence of interest in neural networks. First, the field gained credibility through the efforts of some physicists. They drew analogies between the mathematical behavior of neural networks and spin glasses. Statistical mechanics provided a firmer foundation for some types of neural networks. Second, new and more powerful network architectures were discovered or rediscovered. These architectures addressed the limitations cited by Minsky and Papert. Third, results from neurobiology suggested new architectures for neural networks. Fourth, the availability of cheap and powerful computers allowed widespread experimentation with neural network techniques.

All of these factors led to an explosion of research in the field. A host of international conferences have been held since 1988; and the IEEE in conjunction with International Neural Network Society (INNS) continue to hold the International Joint Conference on Neural Networks (IJCNN) twice a year. The IEEE holds a annual conference each year in Colorado. Annual international conferences are also held in Europe with many smaller conferences and workshops

available. At last count there are five journals dedicated to the field, three newsletters, and some thirty books (with more in publication). The field has attracted a large group of interdisciplinary researchers ranging from neurobiologists and mathematicians to physicists, engineers and psychologists.

III. DEFINITION, CAPABILITIES, AND APPLICATIONS

What is a Neural Network?

While the field is very broad and researchers in the various disciplines use a variety of terms and definitions when discussing neural networks, three characteristics are almost universally accepted as being exhibited by neural networks. Neural networks are collections of simple processing elements connected together and all working at the same time. First, the networks are composed of simple processing elements (neurons). By themselves these elements do not perform any complicated processing. Second, the neurons are connected together into a network topology which allows communication among the neurons. Third, all of the communication and simple neuron processing occurs in parallel. That is to say, all of the computations and communication are done at the same time.

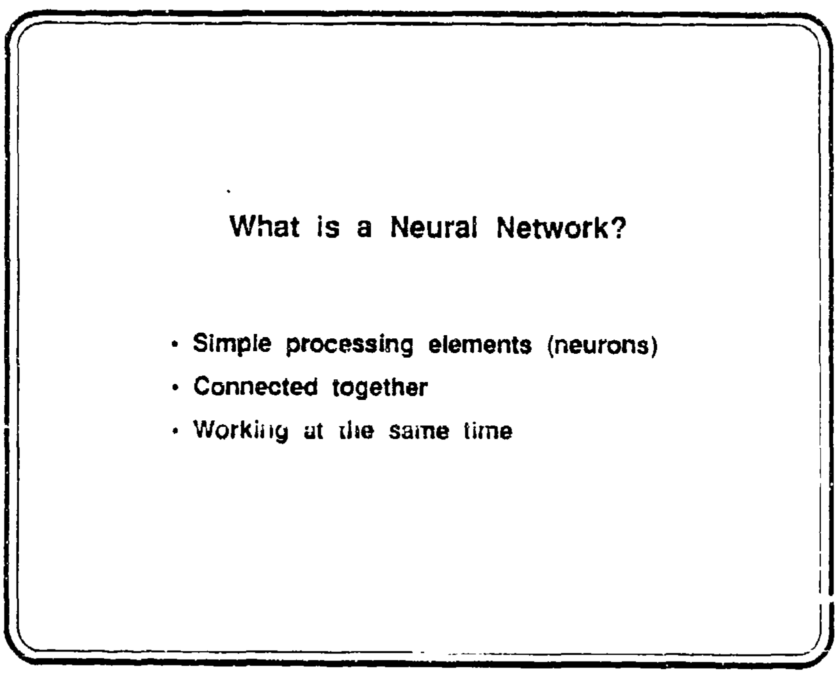


Figure 1. Primary features of a neural network.

Biological Neural Networks

These collections of simple processing elements exhibit some very interesting behaviors. Among these behaviors are pattern classification or recognition, control, adaptation, optimization, and associative memory. We have examples, or existence proofs, of highly connected systems which exhibit the capabilities just mentioned. All of these examples are biological systems—man, dogs, bumblebees, etc.

Capabilities

The most impressive classification and control tasks these biological systems perform are usually taken for granted—recognizing friends, finding food, walking. These are very difficult recognition and control tasks; but they are essential to survival, and biological systems perform them with apparent ease. Despite this apparent ease, these types of tasks have been the most elusive to reproduce using computers or other types of automatons. The hope of neural network research is that neural networks, by emulating some of the structure in the biological nervous system, may better perform these complex tasks.

Components

Biological systems do not obtain their capabilities by employing fast components. In fact, neurons (the basic processing elements of the brain) are extremely slow when compared to the components of digital computers. A typical neuron can process information and produce a response about 100 times per second. Computers can currently produce a response (perform a basic computation) over 50 million times per second. Simply comparing component processing speed, current digital computers operate about 500,000 times faster than a single neuron.

Biological "Computing" Components are *Slow*

Connections or computations made per second

- Neurons: 100 per second
- Computers: 50 million and up

Serial computers are roughly 500,000 times faster than a single neuron.

Figure 2. Operating speed of biological neurons and computer chips.

Brains make up for this speed deficiency by employing many neurons to work on a single problem simultaneously. The human brain has about 100 billion neurons, with each neuron having on average 1000 dendrites (connection paths to other neurons). This allows for a total of 100 trillion synapses or interconnections between neurons. All of these neurons process and communicate information in parallel which means that 10,000 trillion interconnections can be made per second in the human brain. A simpler biological system, the cockroach, can make about 50 billion interconnections per second.

On the other hand, serial computers have one effective interconnection. Even though the fastest computers (Cray XMP-2) utilize this connection 50 million times per second, their overall information processing capability is no match for a biological system. The human brain processes about 20 million times more information than the fastest serial computer, and the "simple" cockroach manages about 1000 times the information of the same computer.

Biological "Computing" Components are *Plentiful*

Human brain

- 100 Billion (10^{11}) neurons
- 1,000 dendrites (connections paths) per neuron
- 100 trillion (10^{14}) synapses (connections)
- All neurons work in parallel
- 10,000 (10^{16}) trillion interconnections per second

Cockroach

- 1 billion (10^{9}) synapses (connections)
- 50 billion interconnections per second

Serial computers

- 1 connection
- 50 million interconnections per second (Cray XMP-2)

The human brain is about 20 million times "faster" than a serial computer

The cockroack brain is about 1,000 times "faster" than a serial computer

Figure 3. Scale of biological neural networks.

Lesson

The main point of these comparisons is just this: Highly interconnected assemblies of simple processing elements produce interesting and useful behavior. When operating in parallel, these same assemblies perform some operations much more quickly than serial computers. In general, these slow,

error prone units (neurons) can perform tasks the fastest computers cannot currently approach. While current neural network research is a long way from reproducing the capabilities of biological systems, the desire to attain and understand these capabilities is the impetus behind much of the research in the field.

Lesson

Highly interconnected assemblies of simple processing elements produce interesting and useful behavior.

These same assemblies perform some operations much more quickly than serial computers.

Figure 4. Lessons from biological neural networks.

Artificial Neural Network (ANN) Implementations

Currently these parallel biological structures are usually simulated on serial computers. Since the simulation is not truly running in parallel, these simulations are very slow compared to their biological counterparts. (Naturally occurring neural networks gain their speed through parallel operation.) Some use is currently made of digital signal processing chips to speed the computations, but this is still essentially a serial process.

Hardware implementations of neurons and networks are currently under development. Several groups (Intel, TRW, the Jet Propulsion Laboratory, AT&T, Syntonics, and others) are building neuron and neural network VLSI chips, and a few of these are actually in production. These chips hold the promise of speeding neural network simulations by several orders of magnitude. Still, problems inherent in the highly connected nature of the problem have yet to be fully addressed.

Areas of long-term research include optical computers and biological computers. Optical computers hold the promise of extremely fast operation and vast numbers of interconnections. Biological computers, if they are fully developed, should allow very dense, small scale constructions. Both of these areas (particularly biological) are many years from commercial implementation.

ANN Capabilities

Associative Memory

Despite these constraints, interesting results are being obtained with simulators in areas such as associative memory or recall. Associative memory involves the completion of a pattern or set of information. This type of memory is typical of human memories. For example, during the course of a conversation someone may discuss an acquaintance who is married, lives in the suburbs, and is a banker. You might be reminded of your uncle John who also has all those characteristics. This type of memory is a two-way street. When discussing your uncle John, you always have in mind that he is a married banker living in the suburbs. A very similar process is pattern completion. The example demonstrated in Figure 5 involves the reconstruction of a complete image of a plane from an occluded view of the plane. Neural networks perform these pattern completion functions as a natural result of their operation.

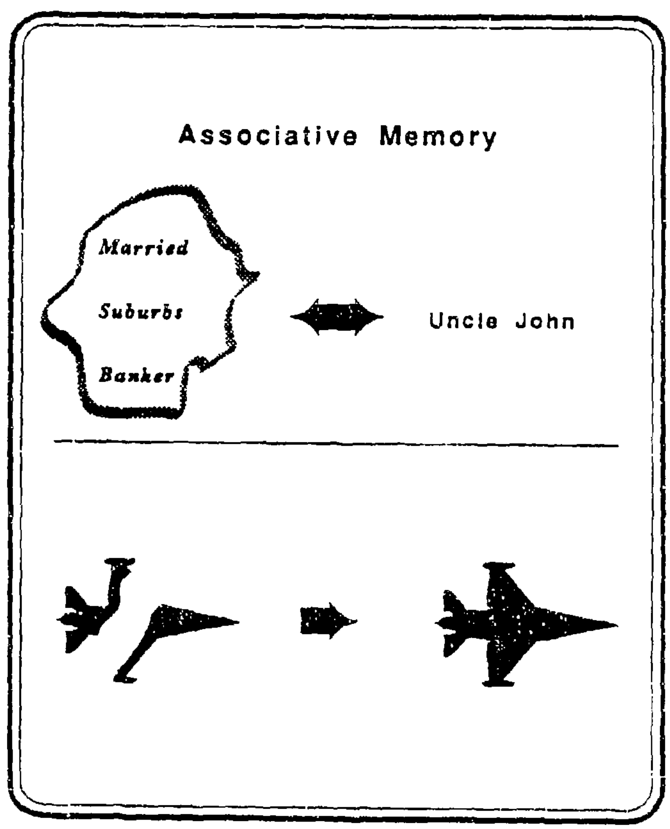


Figure 5. Examples of associative memory.

Pattern Recognition and Classification

As displayed in Figure 6, Another area where natural and artificial networks excel is pattern recognition and classification. As mentioned before, this type of classification is critical to survival of any species and also appears to form the basis of many higher-level cognitive functions. Before any

action can be contemplated, the situation or problem must be recognized and to a certain extent classified.

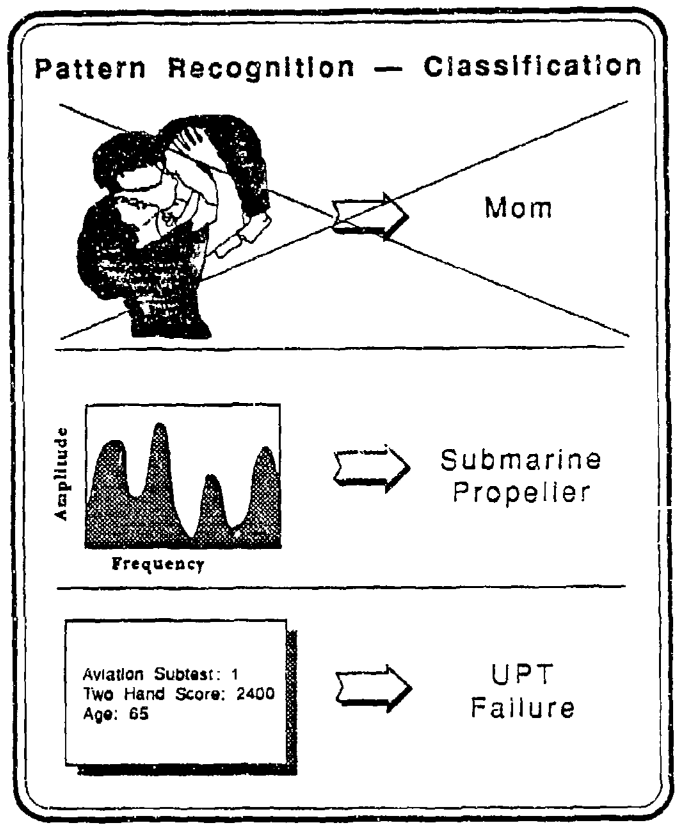


Figure 6. Examples of pattern classification.

In the first example from Figure 6, artificial neural networks are not yet capable of consistently recognizing a picture of mom. Still, this type of difficult classification is an active area of research and several projects are directed at this type of goal.

A second area where pattern recognition is important involves classifying the source of a sonar or radar signal. Given the power spectrum from a sonar signal, the neural network determines whether the source is a submarine propeller or a fishing trawler. This type of classification has met with some research success and is being pursued by several research groups.

The third example in Figure 6 is from the manpower area. Given a candidate's characteristics (test scores, demographic factors, etc.) what is the likelihood the candidate will pass Undergraduate Pilot Training? A similar question could be asked about reenlisting or separating from military service. Many other behavioral models can be cast as classification tasks. Neural networks are also being applied in the areas of optimization, speech, and vision.

Neural Network Applications

Figure 7 displays some examples of specific neural network applications gauged by their stage of development. Those applications classified as in the research stage are typically software demonstrations of a concept. They would involve the solution of small, "toy world" problems using software simulations of neural networks. Applications in the demonstration stage usually have some specific hardware to support the operation of the neural networks and are applied to a larger scale problem representative of the actual area being researched. Fielded systems are either commercial products or completed neural network systems being applied to actual problems. Noteworthy among those applications in this group, credit risk assessment is a commercial product operating entirely in an MS-DOS personal computer environment. This specific application involves classification of credit card applicants according to their credit risk. The problem is similar to many manpower classification problems where a decision must be made based on an individual's characteristics and environmental or economic factors.

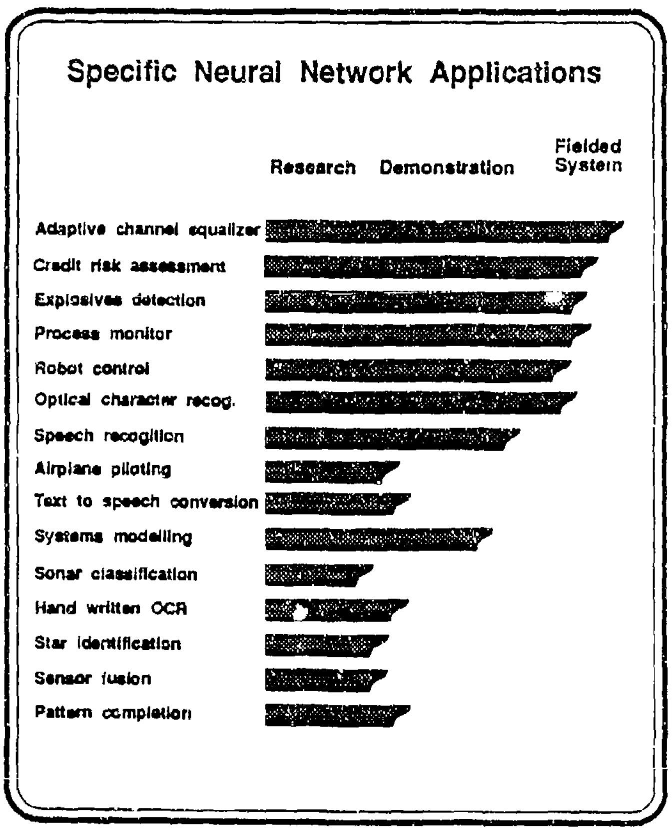


Figure 7. Current status of some neural network applications.

IV. NEURONS AND TRADITIONAL METHODS

Solving Classification Problems

It can be seen that many of the applications in Figure 7 involve solving classification problems. Let's look at how classification problems are usually approached using more traditional techniques. Most of the traditional techniques are parametric, such as regression or logit analysis. These involve the estimation of parameters that divide a decision space (usually binary) based on the inputs or determinants of the classification. Other techniques, clustering algorithms, may not require the estimation of parameters. Instead, these algorithms classify the observed cases into groups whose characteristics are similar.

Each technique provides its own perspective on a problem and has its own limitations. Parametric techniques require the underlying functional form of the relationship be known and specified in advance. An error distribution must also be specified. Clustering algorithms can be even more restrictive in imposing the specific type of a cluster being searched for: nearest neighbor, minimum spanning tree, etc. In general, neural networks impose fewer assumptions about the structure of the problem and thus allow more flexibility in searching for a solution.

Logit Analysis

Figure 8 shows a traditional classification technique, logit analysis, as a black box. Using the individual reenlistment decision as an example, an individual's characteristics are seen as inputs (the grey region on Figure 8). These inputs are fed into the black box which produces an output that is interpreted as the probability the individual will reenlist.

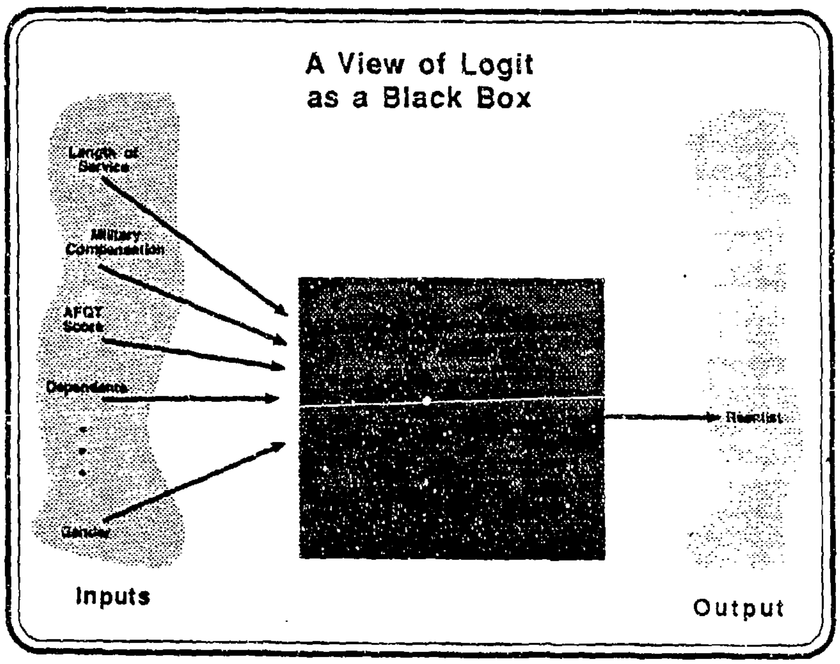


Figure 8. Logit analysis viewed as a black box.

Looking at the problem in more detail (Figure 9), we can see that each of the inputs (I) has a coefficient associated with it, W_1 through W_N. When solving for the probability a specific airman will reenlist, the value of each characteristic for that airman is multiplied by its respective weight. For example **Length of Service** is multiplied by W_1 and **Dependents** is multiplied by W_4. These products are then summed to produce **s** as shown in the left half of the circle in the figure. If we stopped here, this would simply be a linear equation. However, in logit analysis we are looking for a result between zero and one. This result can then be interpreted as a probability of reenlistment. So, the linear sum **s** is passed through a nonlinear transfer function, the logit function, which constrains the output (**a**) to be between zero and one. The logit curve as a function of the linear sum **s** is written in the right half of the circle, and its graph is shown in the box labeled **Logit Function** on the arrow exiting the circle. As can be seen, the logit function transforms the sum s, which may range between positive and negative infinity, to a value between zero and one. As mentioned above, this result is then interpreted as the probability the airman will reenlist.

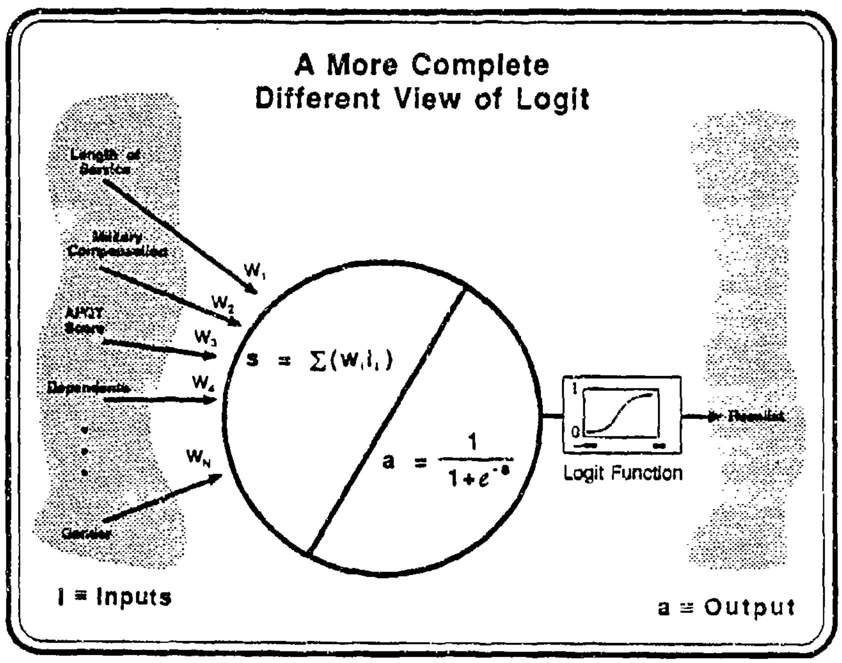


Figure 9. Schematic of logit analysis.

In logit analysis the coefficients W_1 through W_N are chosen by presenting all of the known results to the algorithm. That is, the characteristics and actual result (reenlist or separate) for each airman are presented to the algorithm. The set of coefficients which maximizes the likelihood that the actual decisions would have been observed is then chosen. This usually involves the application of a second-order "hill climbing" technique, such as Newton's method, with likelihood as the objective.

An Artificial Neuron

In Figure 10, we are looking at a typical artificial neuron, again as a black box. As with logit analysis, an individual airman's characteristics are shown on the left. The neuron processes the inputs to produce a predicted reenlistment probability. At this level, the neuron performs the same function as logit analysis. Terminology is the only difference. In this case, the process of converting the inputs into a reenlistment probability is referred to as feed-forward mode. The actual output of the neuron, which we interpret as a reenlistment probability, is referred to as the activation of the neuron.

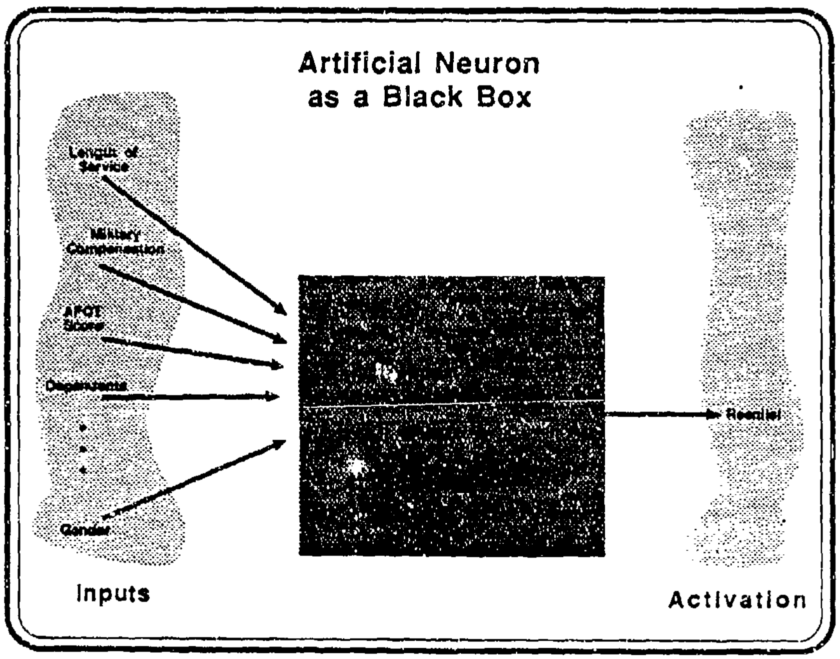


Figure 10. An artificial neuron viewed as a black box.

Figure 11 shows a the computational details for a typical artificial neuron. Again, this figure looks almost identical to the Figure 9. In fact, in feed-forward or prediction mode, the two operate in exactly the same manner. The coefficients W_1 through W_N from the logit analysis are now referred to as weights. The logit function is now called a sigmoid activation function. This is merely a change in terminology, the two functions are identical. If the weights for the neuron could be chosen properly, it would implement logit analysis. The key difference in the neural network paradigm entails applying several neurons to the same problem.

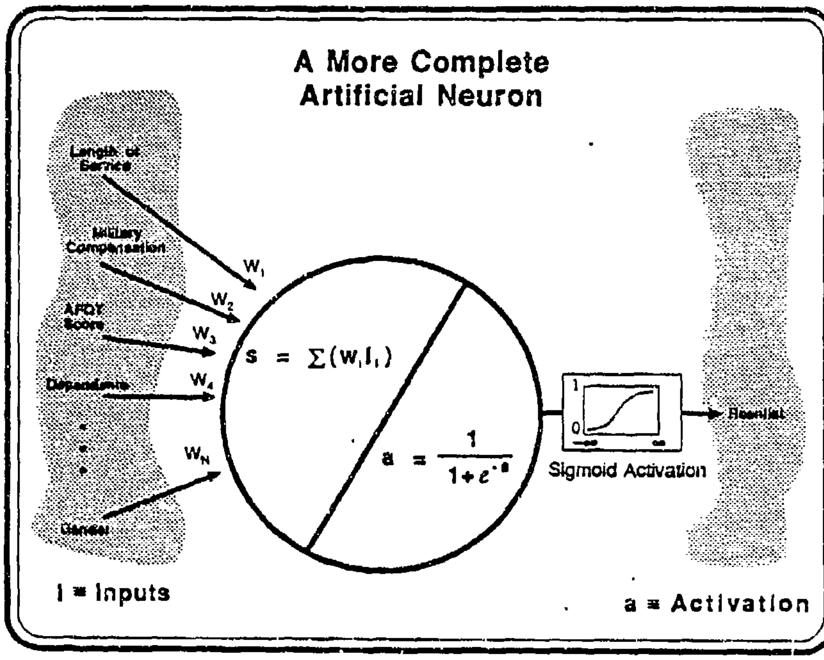


Figure 11. Schematic of an artificial neuron.

V. TWO ARCHITECTURES AND THEIR TRAINING

Widrow-Hoff Learning

The method a neuron employs to determine its weights has yet to be discussed. We will start with a fairly straight-forward method known as Widrow-Hoff or Least Mean Square (LMS) learning. The process of choosing and adapting weights in a neural network is typically referred to as learning. We will continue to use the reenlistment example; and, to simplify the exposition, only two inputs (length of service and number of dependents) will be used (see Figure 12).

As the name implies, the goal of the learning procedure is to minimize the squared error of the predicted reenlistment probability. Training (the process of applying the learning procedure) proceeds as follows. The **Length of Service** and **Dependents** for an airman are presented to the neuron and the neuron produces a guess at the probability of reenlistment. This guess is compared against the actual outcome for the airman (reenlist or separate) and the neuron adjusts itself to produces a better guess (closer to the actual decision).

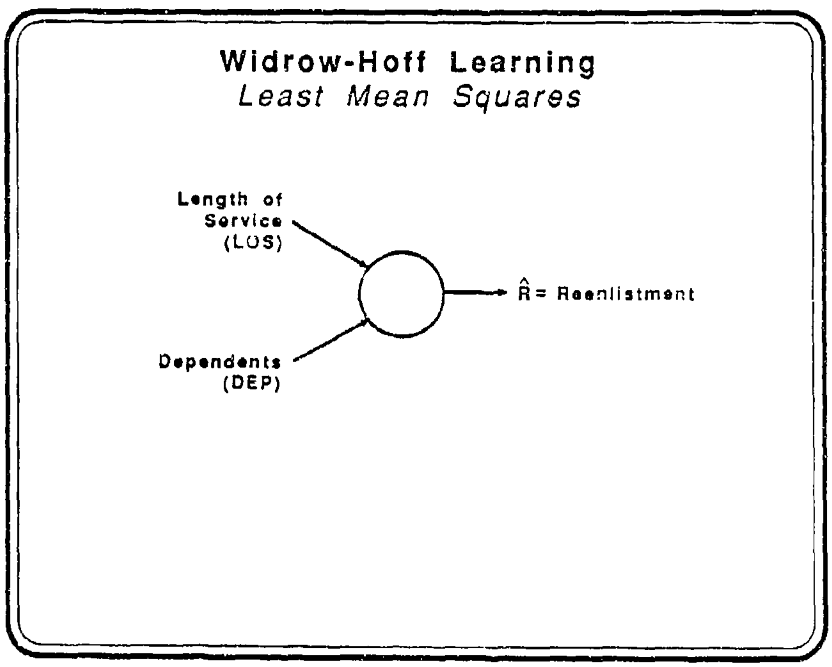


Figure 12. A simple feed-forward network capable of Widrow-Hoff learning.

The process can be seen more explicitly in Figure 13. This neuron produces its guess by taking the product of W_1 with **Length of Service** and summing this result with the product of W_2 with **Dependents.** As can be seen on the **Sum is Activation** line in the figure, this neuron simply forms a linear function of its two inputs. If the goal were to correctly predict this one decision, either W_1 or W_2 could be adjusted to completely correct the prediction. However, the goal is to minimize the squared error over all of the airmen in a sample. Toward this end, the weights are adjusted by the **Weight Adjustment** equations in the Figure 13. A factor known as the learning rate (a small number between zero and one) is employed to determine the distance the weights move relative to the error of a given prediction. If the learning rate is small enough, this learning rule actually implements a first order gradient descent or hill climbing algorithm with the sum of square errors as the optimization criterion. Note that this is an adaptive process and many passes through the sample data are required before the weights converge.

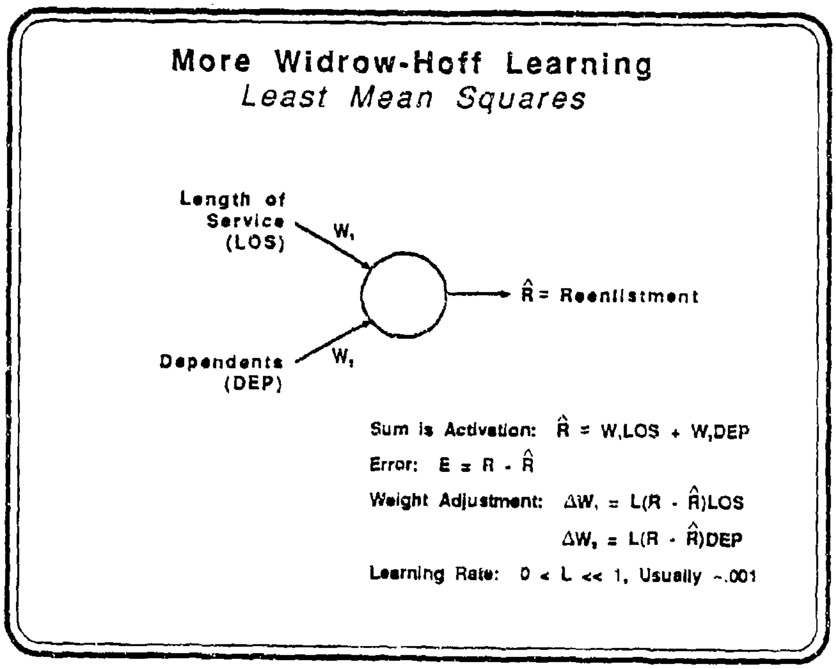


Figure 13. Computations for Widrow-Hoff learning in a feed-forward network.

Convergence

Given the goal (minimum squared error) used in the learning procedure, it should come as no surprise that the weights produced by the Widrow-Hoff method approach ordinary least squares (OLS) regression coefficients. If the two methods are applied to the same data, the Widrow-Hoff weights (provided the learning rate is small enough) will asymptotically approach the coefficients produced by OLS regression. This can be seen in the Figure 14 which represents actual results from a sample of 500 airmen making reenlistment decisions. Starting from zero (the initial choice of weights in Widrow-Hoff learning is irrelevant), the Widrow-Hoff weights move toward the OLS regression coefficients shown at the bottom of the columns. After ten complete passes through the entire training set (all observations), the Widrow-Hoff weights have the same signs as the OLS coefficients. As training proceeds, the Widrow-Hoff weights draw continually closer to the OLS coefficients.

Convergence of Widrow-Hoff Learning

Passes Through Data	W_1 Length of Service	W_2 Dependents
0	-.000	.000
10	-.033	.163
20	-.186	.274
30	-.321	.366
•		
•		
•		
50	-.533	.505
100	-.852	.693
150	-1.001	.769
200	-1.070	.801
250	-1.101	.814
300	-1.116	.820
OLS Coefficients	**-1.134**	**.832**

Figure 14. Convergence of Widrow-Hoff learning.

Characteristics

We can note several characteristics of this learning process. First, as just seen, the Widrow-Hoff weights approach OLS squares regression coefficients. Second, since the process is nearly equivalent to OLS, it can only solve problems which are linearly separable in the inputs. This was Minsky and Papert's complaint in 1969. Third, and again because the neurons are linear, adding

multiple layers to this process adds nothing. Any series of linear sums, can be condensed into a single linear equation. Fourth, and the only difference with respect to regression, the process is adaptive. If the inputs are not stationary (ie. the environment is continually changing), Widrow-Hoff learning can continually adapt the weights to reflect changing patterns in the input-output mapping.

This adaptive aspect is taken advantage of in an application listed earlier, the adaptive channel equalizer. This device searches for the best frequencies to transit information using high speed modems over phone lines. Toward this end, the Widrow-Hoff neuron is the most prolific of current artificial neurons; there is one in every high speed modem.

Comparison of Terminology

At this point it might be useful to compare the terminology of regression analysis and neural networks (at least as it applies the models discussed thus far). The terms on the left of Figure 15 are typically employed in regression analysis and their analogues in neural network terminology are listed on the right. The coefficient vs. weight, output/result vs. activation, and logistic curve vs. sigmoid curve are direct analogues in the two domains. There is also a great amount of similarity in solving a regression equation and the feed-forward processing in a neural network. Likewise estimation and training are processes that work toward a similar end. Stretching the analogies a bit, a neuron (at least at some times) can be considered a function and a neural network composed of neurons can be considered a system of equations.

Comparison of Terminology

Logit/Regression		Neural Network
coefficient	⟷	weight
output/result	⟷	activation
solving	⟷	feeding forward
logistic curve	⟷	sigmoid curve
estimation	⟷	training
function	⟷	neuron
system	⟷	neural network

Figure 15. Comparison of neural network and regression terminology.

Back Propagation

The Training Algorithm

We now proceed to a learning mechanism similar to Widrow-Hoff but more powerful. This method, back propagation, is used in about one-third of current research and perhaps three-quarters of current applications. Sticking with the reenlistment example, the two inputs (**Length of Service** and **Dependents**) can be seen at the far left of Figure 16. Both of these inputs are now fed into two neurons and each neuron produces an activation (output).

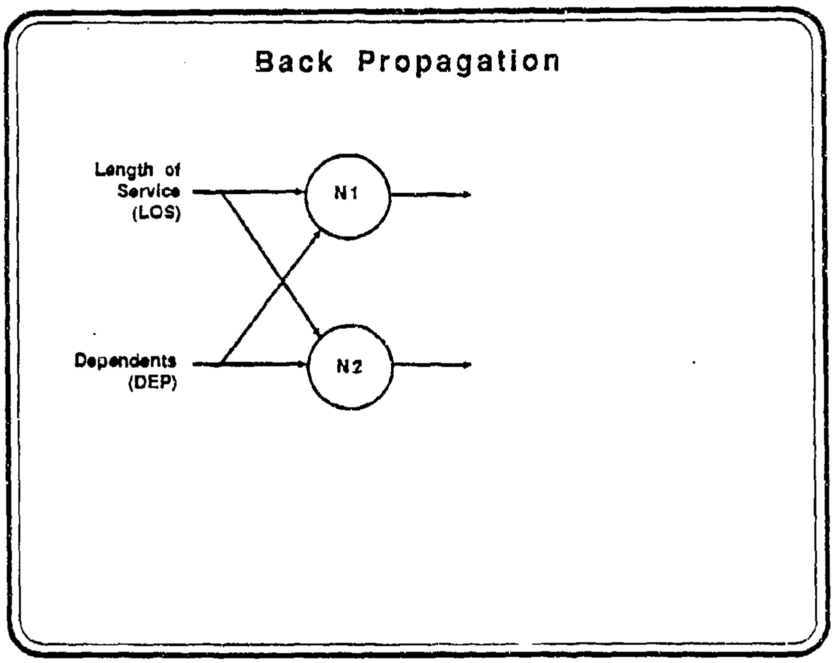


Figure 16. First layer of a simple feed-forward network to be trained by back propagation.

Looking at this process in more detail (Figure 17), these two neurons (**N1** and **N2**) are seen to be identical to the first neuron we examined. Specifically, neuron 1 (**N1**) forms a sum (S_{N1}) which is just a linear combination of the two inputs using the weights W_1 and W_3 to form the combination. This linear relationship is shown in the **Sum** line. The sum is then passed through the non-linear sigmoid activation function to produce an activation (A_{N1}) between zero and one for the neuron. Neuron 2 (N_2) performs exactly the same computation using weights W_2 and W_4 to form the linear combination of **Length of Service** and **Dependents**.

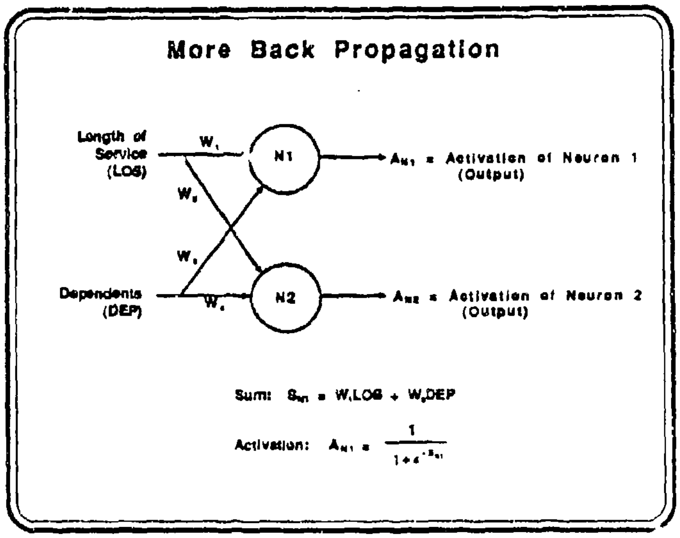


Figure 17. Computations in the first layer of a feed-forward network trained with back propagation.

As seen in Figure 18, the activations or outputs of these two neurons are fed into a third neuron (N_3) which uses them as its inputs. This third neuron produces its activation using the same process as the first two neurons–N_1 and N_2. It sums the products of its two inputs with their respective weights (W_5 and W_6) and passes this linear sum through the sigmoid activation function. The resulting activation or output is then interpreted as a reenlistment probability. We have now put more than one neuron to work on one problem.

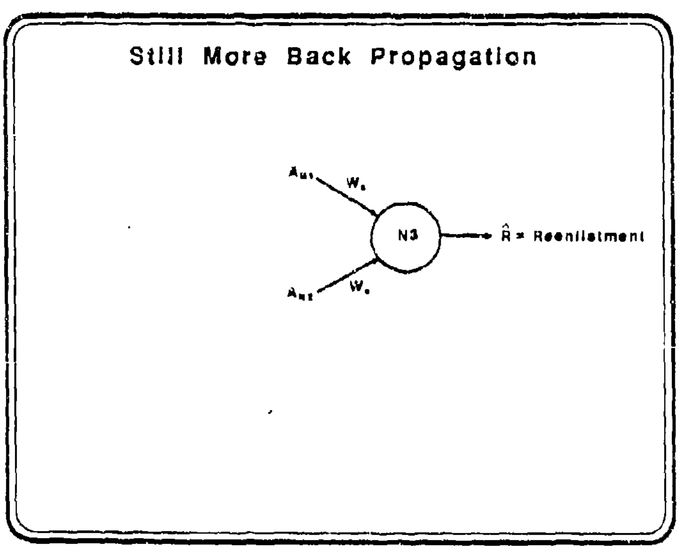


Figure 18. Output layer of a simple feed-forward network.

Putting the entire process together in Figure 19, we can see that the two inputs (characteristics of an airman) feed forward into neuron 1 and neuron 2. These two neurons each produce an activation which is fed forward into the third neuron who also produces an activation. This final activation is interpreted as a reenlistment probability (**R**-hat). The first two neurons are collectively referred to as the hidden layer.

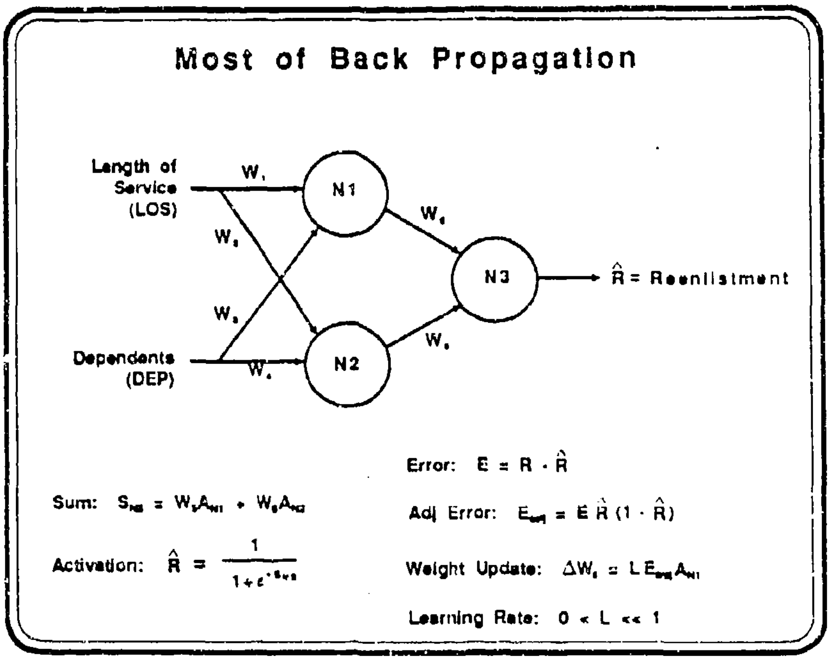


Figure 19. Computations for the output layer of a feed-forward network trained with back propagation.

This brings us to the problem of determining the weights ($\mathbf{W}_1$ through $\mathbf{W}_6$) used in the feed-forward process. Back propagation uses a variant of the Widrow-Hoff rule presented earlier to adapt the weights in the network. Again, the goal is to minimize the squared error of the predicted reenlistment probabilities over all of the observations (airmen in this case). The weights normally start the training process as small random values. Their actual values are unimportant, but training will fail if they all start at the same value. The characteristics of the first airman are applied to the network; and, through the feed-forward process described in the previous two figures. Using the random starting weights, a predicted reenlistment probability (**R**-hat) is produced.

As shown in the **Error** equation, this predicted probability is compared against the airman's actual decision (separate = 0, reenlist = 1) to produce an error for the prediction. This error is then used to adjust the weights on neuron 3 as shown in the **Adj Error** and **Weight Update** equations. As in the case of Widrow-Hoff learning, this process just implements a first order gradient descent method with the sum of squared errors as the criterion. The computations are somewhat more involved because of the neuron's nonlinear transfer function. The **Weight Update** equation shows the change in weight 5; weight 6 is adjusted in exactly the same manner with $\mathbf{A}_{N2}$ substituted for $\mathbf{A}_{N1}$ in the equation.

Thus far, the learning process can only adjust weights 5 and 6 as neuron 3 is the only neuron for which an error can be directly calculated. This is known as the credit assignment problem. To assign credit, the third neuron essentially places some of the blame for its error on the two neurons

who supplied it information (N1 and N2). It propagates some of its error back to the two neurons in the hidden layer (hence the name back propagation). The third neuron uses the weight connecting it with the hidden neuron to back propagate this error. As seen in Figure 20, the error for N_1 is just the adjusted error (E_{adj}) computed earlier for N_3 multiplied by the weight (W_5) connecting the two neurons. Now that N_1 has an error, it can adjust the weights on its inputs (W^1 and W^3) in precisely the same manner the third neuron adjusted its weights. The second neuron (N_2) adjusts its weights likewise (using W_4 to compute its error). This whole process utilizes the chain rule of derivatives to perform first order, gradient-descent search in weight space.

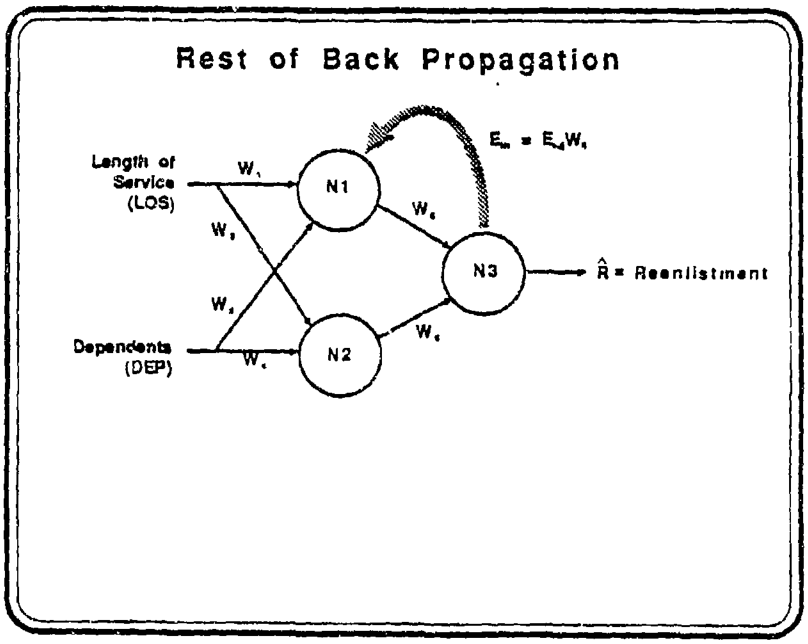


Figure 20. Backward propagation of the error signal in a feed-forward network.

The rule was developed by Paul Werbos in his 1972 Harvard Dissertation thesis *Beyond Regression. New Tools for Predictions and Analysis in the Behavioral Sciences* and was subsequently rediscovered in 1984 by David Rumelhart, Geoffrey Hinton, and Ronald Williams.

Back Propagation Capabilities

The back propagation method contributed greatly to the resurgence of interest in neural networks for one primary reason. It overcomes the problems of linear neurons described by Minsky and Papert in 1969. Specifically:

> Feed-forward neural networks (trainable by back propagation) with non-linear transfer functions (sigmoid activation functions) and one hidden layer (**N1** and **N2**) can approximate any arbitrary continuous function of inputs.

This capability is critical, because it means a feed-forward net can be used as a universal function approximator.

> *Feed forward neural networks with non-linear transfer functions and one hidden layer approximate any arbitrary function of inputs.*

Figure 21. Unique capability of a feed-forward network.

VI. APPLYING NEURAL NETWORKS

Some Example Applications

"Universal" Approximation with Back Propagation

To demonstrate this capability, we will work with a well-behaved (but highly non-linear) function—the saddle curve. The X-Y-Z triplets listed on the left of Figure 22 represent points on the saddle curve. The function which provides the Z value for any X-Y pair is shown in the center of the figure.

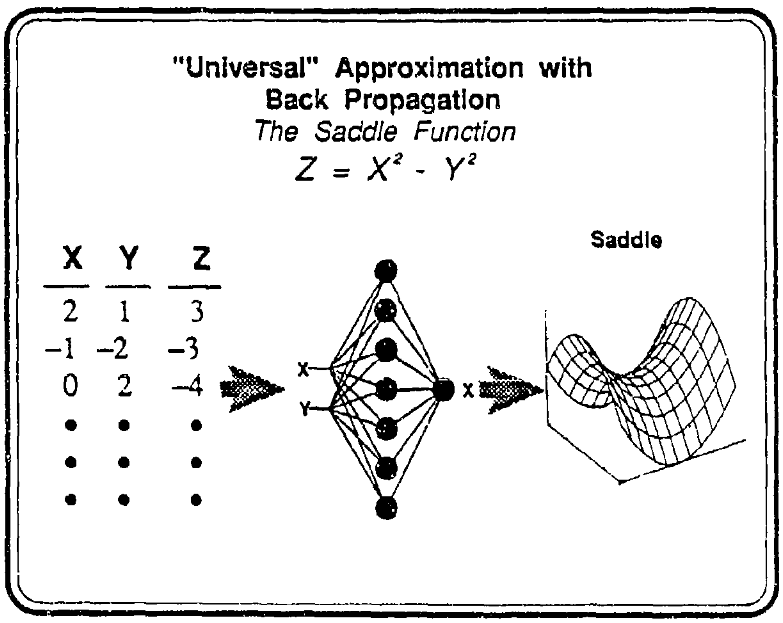


Figure 22. Approximating the saddle function with a feed-forward network trained with back propagation.

The X-Y-Z pairs are provided one at a time to a neural network which uses the back propagation method to train itself to the inputs. The network is shown at the center of the figure. In this case there are eight hidden neurons instead of the two from the earlier example. The network takes a single X-Y pair and feeds it forward through the 8 hidden neurons and then the output neuron to produce a guess at the Z value. The actual Z value is compared to the guess and the network weights are adjusted as discussed earlier. Multiple training passes are made through all of the X-Y-Z pairs until the network converges and the weights cease to change.

After training is completed, a set of X-Y pairs are presented to the network which then produces "guesses" at the corresponding Z values. If we plot these "guesses," as we have at the right of Figure 22, we see that the network has learned to reproduce a saddle function. It has found this form with no prior hints as to the structure of the underlying function.

Looking at a different function, the hat function, the same experiment can be performed. Generate a set of X-Y-Z triplets on the function. Train the network to this data. Supply the network with X-Y pairs. Let the network predict Z values and plot the results. We can see that the network now reproduces the hat function. It is important to note that this could be exactly the same network we started with when the saddle function was learned. In fact, we could have taken the final network which reproduced the saddle function and trained it to the hat function data triplets. The network would have "forgotten" the saddle function and learned to reproduce the hat function.

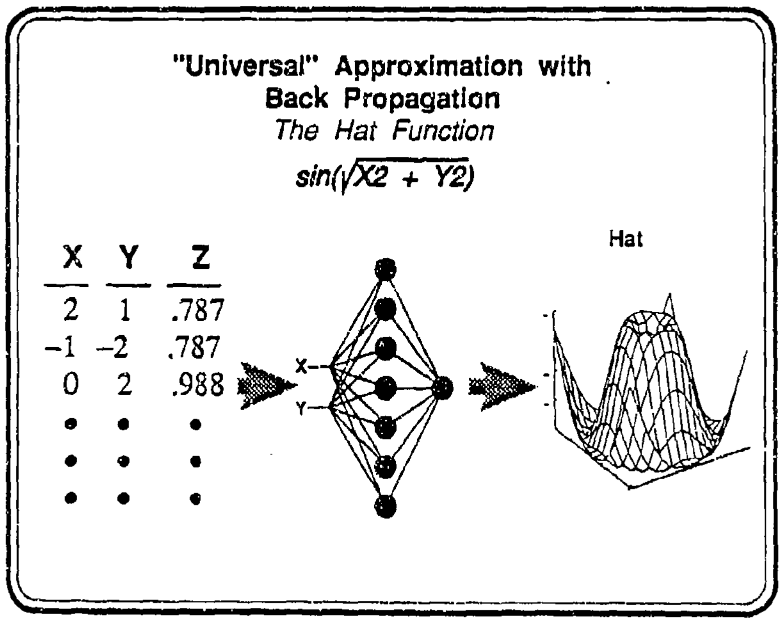


Figure 23. Approximating the hat function with a feed-forward network trained with back propagation.

Back Propagation and Reenlistment Decisions

We can demonstrate the flexibility of back propagation on some actual data. For the sake of exposition, and to keep the problem easy to visualize, we will again look at enlisted airmen reenlistment as a function of length of service and number of dependents. The data for this example is taken from the actual decisions of all Air Traffic Controllers from fiscal year 1976 to fiscal year 1986. The plot labeled **Logit** in Figure 24 represents the separation of the airmen into reenlisters and separators by logit analysis, given only length of service and dependents as inputs. Those airmen in the dark grey region are classified as reenlisters by logit analysis and those in the white are classified as separators. The graph demonstrates that logit forms strictly linear classification boundaries of its inputs. As noted, logit correctly classifies 64.2% of the decision makers. This means that 35.8% of those in the dark grey region actually separated or those in the white region actually reenlisted.

The second plot shows the classification regions found by back propagation on the same data. As can be seen, the regions formed are much more complex (back propagation can form arbitrarily complex classification regions). Back propagation correctly classifies 69.5% of the decision makers which appears to be marginal improvement over logit. The real differences between the two approaches becomes apparent if we divide the input data into cohorts.

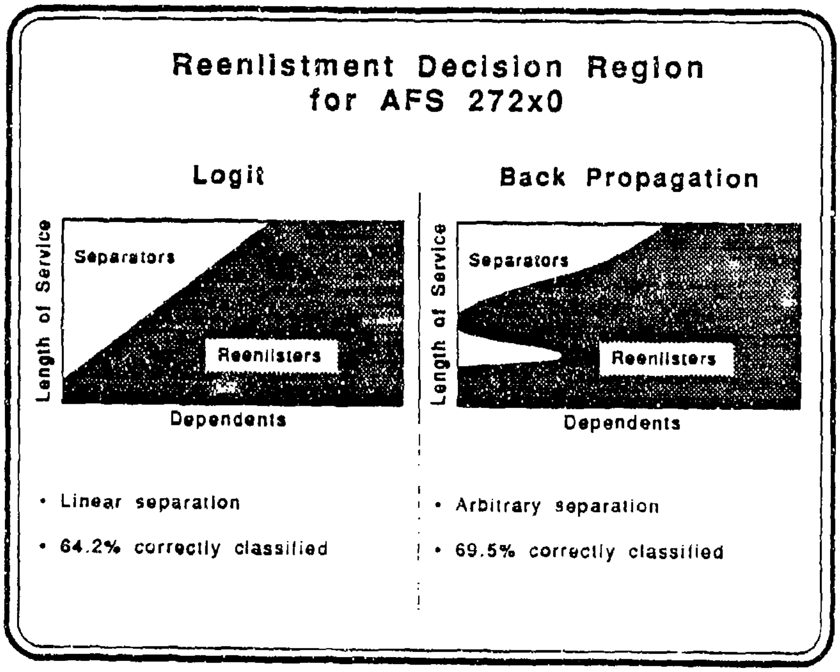


Figure 24. Reenlistment decision regions formed by logit analysis and a back propagation network.

Figure 25 displays the actual and predicted (logit and neural network) reenlistment rates by cohort. The cohorts are based on a two-way separation across number of dependents and length of service (in years). For example, the actual reenlistment rate for air traffic controllers with four years of service and one dependent was .33; that is to say, one third of these airmen reenlisted. It can be seen in the table that the predicted reenlistment rates from the neural network are much closer to the actual rates than those produced by logit analysis. The average predicted error by logit analysis across all of the cohorts is .093 while the average error for the neural network is .02. In fact, the largest cohort error for the neural network is only .07 while in the fifth year of service with one dependent logit analysis is off by .32. Looking at the critical fourth year of service, when most decisions are made, the average cohort error for logit is .15 while the neural network is always within .02 of the actual rate.

Comparison of Logit and Neural Network Predicted Reenlistment Rates by Cohort

Length of Service	Number of Dependents 0	1	2	3	4
	Actual Rates				
3	.56	.69	.75	.82	.93
4	.17	.33	.46	.58	.70
5	.52	.67	.75	.75	.82
	Predicted Rates using Logit Analysis				
3	.47	.61	.73	.83	.91
4	.34	.47	.61	.74	.84
5	.23	.35	.48	.62	.74
	Predicted Rates using Neural Network				
3	.57	.68	.76	.82	.86
4	.19	.32	.46	.59	.70
5	.51	.61	.72	.80	.83

Figure 25. Reenlistment projections of logit and back propagation by cohort.

This comparison can be made even more dramatic by plotting the reenlistment rates by cohort. Looking at the graph in the upper left hand corner of Figure 26, the length of service and number of dependents form the bottom two axes with the reenlistment rate plotted on the upright axis. The surface shown follows the actual reenlistment rates observed for air traffic controllers. As can be seen in the middle graph, logit analysis misses the shape of the relationship altogether. Without prior knowledge of the structure of the relationship, which would have to be encoded into the input variables, logit was doomed to form a near-planer relationship. On the other hand, as seen in the bottom graph, the neural network captures the relationship almost perfectly.

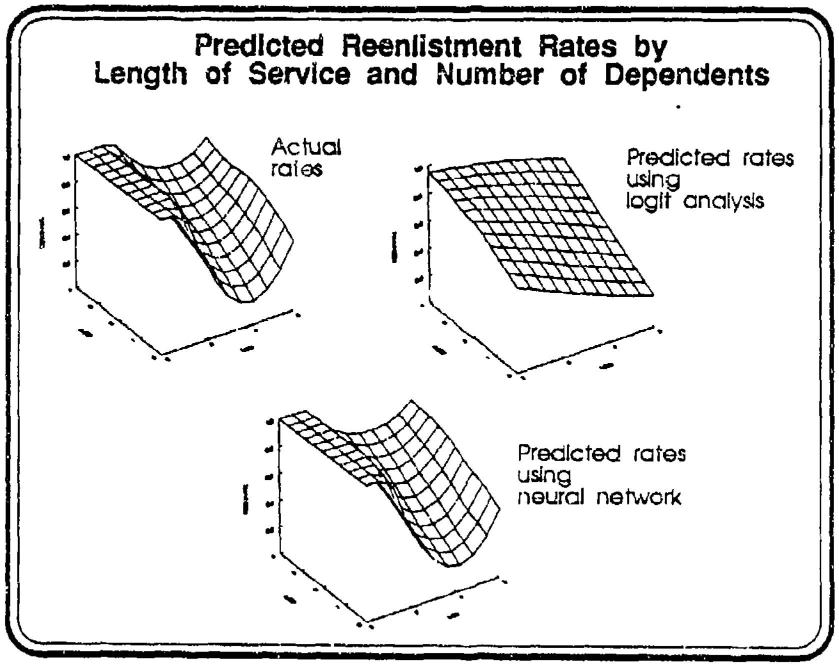


Figure 26. Reenlistment rate surfaces produced by logit and back propagation.

Classification Capabilities

In general, the complexity of the decision region a feed-forward neural network can form depends on the number of layers of neurons in the network. A single layer, as depicted at the top of Figure 27, forms a linear decision region (or hyperplane in the case of multiple inputs). In effect, a single neuron performs a very close analogue of a logit analysis. A network with two layers of neurons, as seen in the middle of the figure, can form any convex surface. As shown at the bottom of the figure, a network with three layers of neurons can form any arbitrary decision region. The region can form any shape and even be disjoint. It is also possible for two layer networks to form non-convex and discontinuous surfaces. However, they are not guaranteed to be capable of reproducing any possible non-convex or discontinuous surface.

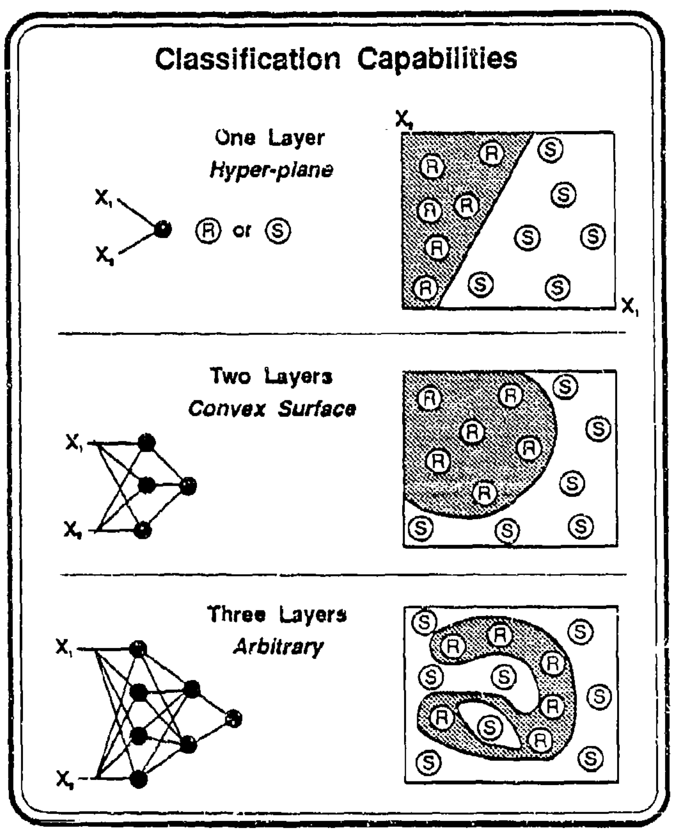


Figure 27. Classification capabilities of feed-forward neural networks.

Steps in Applying Neural Networks

Continuing with the analogy between neural networks and regression analysis, we can compare the normal steps involved in carrying out an analysis using these techniques. (In the case of neural networks, we are really only addressing a subset neural network architectures.) As seen in Figure 28, both techniques require the identification and categorization of the output. What are we interested in producing and is it binary, continuous, cardinal, or some other form? Identification of

the inputs or determinants is also required by both techniques. Only regression requires specification of an exact functional model. To a certain extent the number of neurons and layers of neurons in the network determine the complexity of the relationships it can capture. However, this is not nearly as stringent a requirement as development of a specific functional form. Both techniques also require the estimation or training of the model. In both cases, it is also common to validate the resulting model against data not used during training or estimation. In the case of regression analysis, it is usually possible to evaluate the statistical significance of the estimated parameters, assuming the errors follow some well-defined and specified distribution. The primary differences are the inherent flexibility of the neural network and the inability (in general) to test the statistical significance of a neural network model.

Steps in Applying Neural Networks and Regression Analysis

	Regression	Neural Network
1. Identify required output.	✓	✓
2. Identify inputs or determinants.	✓	✓
3. Develop a specific functional model.	✓	
4. Estimate or train the model to known observations.	✓	✓
5. Evaluate the statistical significance of the model's parameters	✓	
6. Validate the model against data not used during training	✓	✓

Figure 28. Comparison of the steps in applying neural networks and regression analysis.

Other Architectures

In this primer, we have really only addressed two specific neural network architectures (Widrow-Hoff and back propagation) and these two are highly related (both use error correcting learning methods). The network architectures that have been documented in the literature number somewhere in the 40s with many architectures having multiple variants. Some employ slightly different neurons and there are many approaches to learning. While the two networks discussed here barely scratch the surface of a very broad field, most other architectures share major features with the two discussed and all share the three characteristics outlined at the beginning. They all use simple processing elements connected into a network and all working in parallel.

An Example Back Propagation Training Path

Figure 29 demonstrate some stages in learning of the saddle function discussed earlier. The network's process of adapting to the saddle function can be clearly seen in the progression of training snapshots. For this example, a back propagation network with twelve hidden neurons in one layer was trained on a set of 120 evenly spaced X-Y-Z triplets. The X-Y pairs were evenly distributed between -2 and 2 which causes the Z value to vary from -4 to 4. (A linear transfer function was used on the output neuron to allow for a range beyond 0 to 1.) The graph in the upper left corner shows the Z values produced by the network for each X-Y pair before any training has occurred. Essentially, these Z values are just small random values centered around zero. The graph (after 35 training runs) shows a more uniform Z surface, but still does not look much like the saddle function. After 40 training passes through all of the X-Y-Z triplets (lower left corner), some shape begins to emerge. And, after 90 training runs, the network is reproducing the saddle function with very little error. This ability to adaptively capture underlying relationships directly from observed behavior is one of the primary capabilities of neural networks.

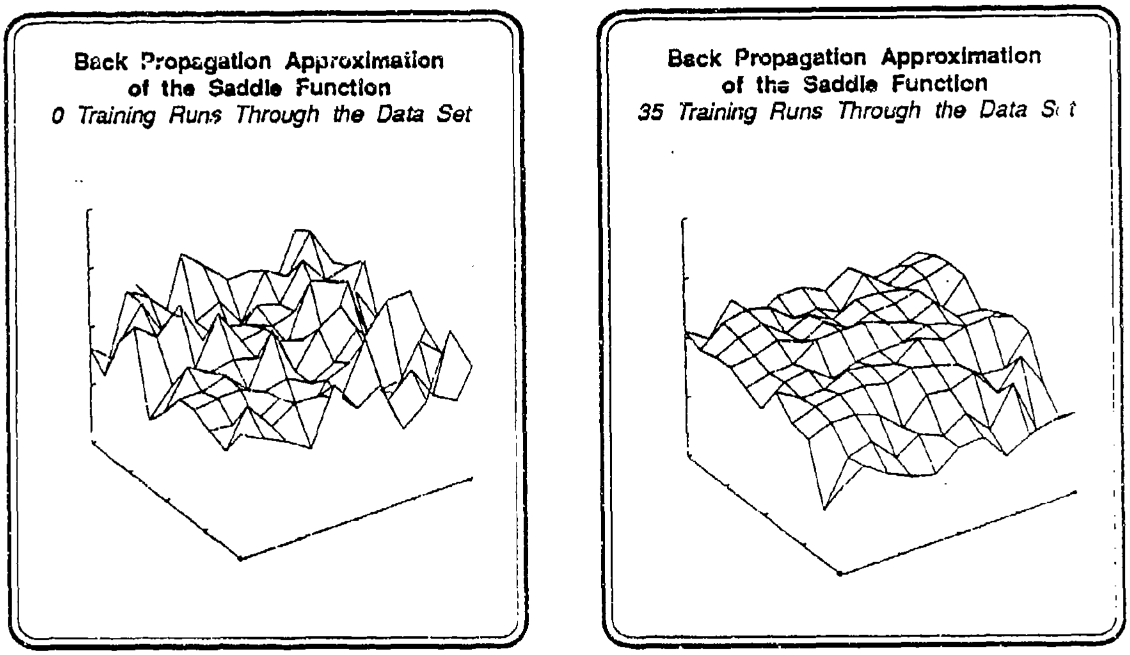
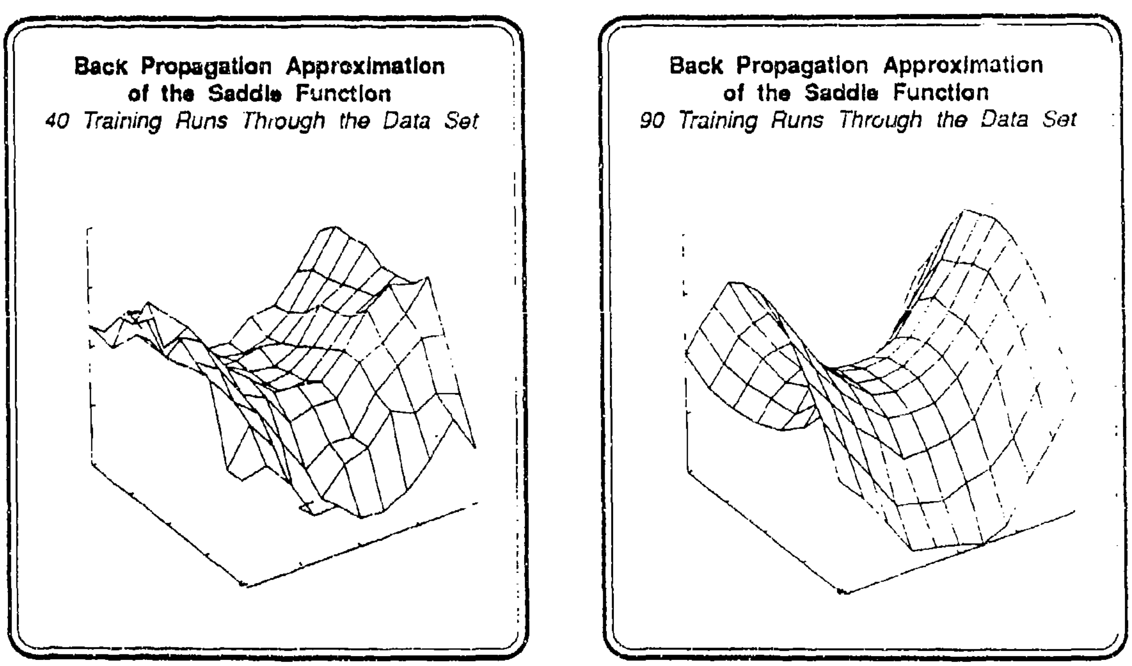


Figure 29. Back propagation networks performance on the saddle function at various stages of training.



VII. CONCLUSION

While the field is still relatively young, artificial neural networks currently offer personnel researchers and policy makers several powerful facilities for analyzing both individual decisions and group behaviors. As seen in the elementary reenlistment example, the ability of networks to extract nonlinear information from noisy and conflicting examples of individual actions can allow researchers to better model complex behavior patterns. Particularly in areas traditionally addressed by regression and other functional based techniques, neural networks provide a more flexible format for model development. Several network architectures allow the underlying and potentially nonlinear form of the model to be determined directly from the observed behavior of a system or sample of individuals. This ability should prove important in personnel analysis and lies at the heart of the recent success neural networks have demonstrated in areas ranging from sonar classification to system control. While artificial neural networks will not supplant traditional statistical methods in the near future, they provide some very powerful alternate techniques for data analysis and model building.